Chapter 5: Gene-Gene Interaction, Pathways and Classification of Acid-sensing ion channels (ASIC) genes in pain and non pain conditions

5.1 Introduction

Can our genes detect how we perceive and accept pain or what makes neurons sense extracellular acid? What kinds of genes are involved in pain and no pain conditions? These questions have given birth to the increasing investigation on acid-sensing ion channels (ASICs) in the central and peripheral nervous system over a decade because these are directly activated by extracellular acidity, which is a prominent cause of pain. There are lots of proposed genes contributing to nociception and pain perception. These pain genes have been identified and studied in isolation and in groups. Using systems biology techniques, genes can be studied in the context of biological pathways and networks in which they function. Genetic approaches to investigating pain pathways have identified the molecular nature, changing mechanisms in neurons and the role of immune system cells. These techniques of studying gene-gene interaction and the pain pathways have complemented the old traditional neuroscience approaches of pharmacology, electrophysiology to gain insights into pain perception. These gene-gene interactions and their classification cannot be solved by a statistical method due to the large datasets generated and the multiple polymorphisms involved. Hence, it requires informatics approaches such as machine learning methods to analyze and interpret relevant data and to classify the genes into pain and no pain class. In our work we have used machine learning techniques and present the strengths and weaknesses of each machine learning method in

detecting gene-gene interactions related to human pain. Through this work, we have made an attempt to show the genetic viability towards chronic pain.

Pain, either chronic or acute can affect the quality of life. Studying genes that play a role in pain perception can certainly provide a target for developing new treatment and help physicians in better assessment of patient's perceptions of pain. We often find some people having a higher tolerance for pain than others. Ongoing in depth, we find the answer is genetic. Chronic pain may occur due to an ongoing ailment, such as cancer, arthritis or an infection or it can even occur in people who have suffered injury or illness. In the case of acute pain, the sensation is triggered by the nervous system to make the body alert to possible injury whereas in chronic pain we find it to be persistent, with pain signals firing regularly in the nervous system for weeks, months or years. By classifying the pain intensity level of chronic patients undergoing treatment, it was observed that the patients falling into a particular class of pain intensity level do possess genetic similarity too. Symptoms like low back pain, nerve pain or a headache is generally observed in people who suffer from chronic pain. Physicians may take a variety of approaches like drugs, local electrical stimulation, brain stimulation, acupuncture and even surgery may be used in treatment of chronic pain.

The term gene-gene interactions are also known as epistasis and genetic interactions. It also can be defined as a logical interaction between two or more genes that affects the phenotype of organisms. The ultimate goal of gene-gene interaction is to recognize gene function, identify pathways and discover potential drug targets. Moreover, there are various types of gene-gene interactions such as synthetic interaction, epistatic interaction, and suppressive-interaction that are shown in Figure 5.1. These interactions are particularly important due to the effect of a gene on individual phenotype is depending on more than one additional gene.

As shown in Figure 5.1, there are various types of gene-gene interactions. For instances, synthetic interaction between two genes is that genes A and B are on different parallel pathways that can obtain the purple phenotype C. If either of the genes is knockout, the purple phenotype C still can be viewed. However, if both of the genes are knockout, it will result in a nonpurple phenotype. Next, the example of epistatic-interaction that is the wild type holds a mixed purple and green phenotype of genes C and D. A gene knockout of gene B cannot obtain a purple phenotype of gene C, but green phenotype of gene D still can be seen. A gene knockout of gene A cannot obtain the green and purple phenotypes. Furthermore, the example of suppressive-interaction is wild type phenotype showing a purple phenotype since gene A suppresses gene B and gene C is active.

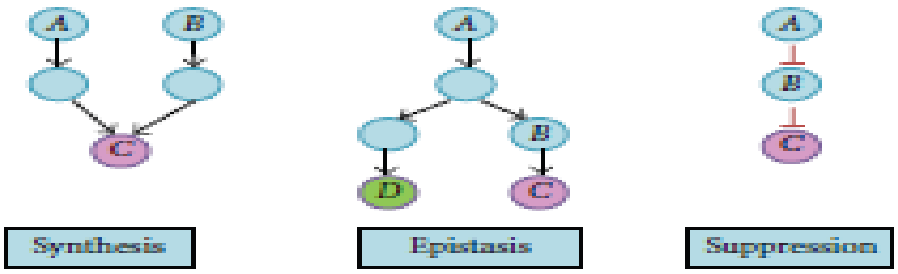

Figure 5.1: Types of gene-gene interaction

A gene knockout of gene B has no effect for result purple phenotype. A knockout of gene A results in a non-purple phenotype since gene B is still suppressing gene C and if both of the genes A and B are knockout will result in wild type phenotype.

Moreover, there are various challenges that are associated with gene-gene interactions that need to be addressed. The greatest challenge is the increasing volume of data that needed to be analyzed. The number of potential interactions increases as the number of SNPs increases. This leads to high computational

complexity because it needs to enumerate all possible SNP combinations in multilocus associations at genome-wide scale. Hence, jointly analyzing such SNP combinations by high throughput genotyping technologies is also one of the challenges faced in genome-wide association studies. Besides, the existence of high dimensionality of data and multiple polymorphisms has also increased the computational complexity of traditional statistical approaches to analyze large-scale genetic data. Hence, the existence of machine learning methods can overcome these challenges because machine learning methods are flexible in recognizing the gene-gene interactions that can contribute to individual's pain status.

We focus on supervised machine learning in which the machine undergoes learning process and predicts the type of gene interactions based on the given inputs. Hence, the goal of supervised machine learning is based on given input variables and then predicts the output variables [McKinney, B. A. *et al*., 2006]. The methods of machine learning that we focus on here are neural networks (NNs), Naïve Bayes classifier, Bayesian logistic regression, rule-based, random forests (RFs), CART and support vector machine (SVM).

The next approach was to investigate the acid-sensing ion channels (ASIC) genes that play a prominent role in pain sensation. Four acid-sensing ion channel (ASIC) genes (ASIC1, ASIC2, ASIC3, and ASIC4) and six ASIC subunits (ASIC1a, ASIC1b, ASIC2a, ASIC2b, ASIC3, and ASIC4) have been identified [Krishtal, O. 2003]. Acid-sensing ion channels (ASICs) are proton-gated Na (+) channels. They have been implicated with synaptic transmission, pain perception as well as mechanoperception. ASIC4 is the most recent member of this gene family. It shows expression throughout the central nervous system with the strongest

expression in the pituitary gland. ASIC4 is inactive by itself and its function is unknown.

5.2 Structure

Although the exact subunit composition (or subtypes) of ASICs in most neurons remains unclear, almost all ASIC subunits are known to be present in primary sensory neurons [Benson, C. J. *et al*., 2002] [Page, A.J. *et al*., 2005]. ASIC1a, ASIC1b, ASIC2b and ASIC3 are extensively expressed in small and medium nociceptive neurons [Alvarez, D.L.R.D. *et al*., 2002]. ASIC2a and ASIC3 are also expressed in medium and large sensory neurons [Price, M.P. *et al*., 2001]. In the central nervous system, ASIC1a, ASIC2a, and ASIC2b are widely expressed in the brain [Jovov, B. *et al*., 2003]. The presence of ASICs other than ASIC1a in the dorsal horn of spinal cord, where pain-related signals relay and transmitted to the brain, is less clear [Wu, L.J. *et al*., 2004]. ASIC4, which cannot be activated by protons, has been detected in the pituitary gland, brain, spinal cord, and retina [Grunder, S. *et al*., 2000].

5.2.1 Role of ASICs in Pain Sensation

Physiological pain is initiated by high-threshold unmyelinated C or myelinated Aδ primary sensory neurons that feed into nociceptive pathways of the central nervous system [Basbaum, A.I. *et al*., 2009; Costigan, M. *et al*., 2009]. The notion that ASICs function as a major sensor of acid-evoked pain is supported by the following evidence: ASICs are expressed in peripheral sensory neurons as well as spinal nociceptive pathways (e.g., spinal cord dorsal horn); different homomeric and heteromeric ASICs are well positioned to detect and differentiate pH variations in both physiological and pathophysiological ranges; and more importantly, inhibiting ASICs has been shown to relieve pain in a

variety of pain syndromes in both animals and humans [Wemmie, J.A. *et al*., 2006; McCleskey, E.W. 1999].

5.2.2 Primary inflammatory pain

Direct perfusion of acidic solutions or iontophoresis of protons into the skin causes pain in humans [Steen, K.H. *et al*., 1995; Ugawa, S. *et al.*, 2002; Jones, N.G. *et al*., 2004]. This acid-evoked pain can be significantly reduced by amiloride [Ugawa, S. *et al*., 2002], a common inhibitor of ASICs, and nonsteroid anti-inflammatory drugs (NSAIDs) such as declofenac and ibuprofen, which selectively inhibit ASIC1a and ASIC3, respectively [Voilley, N. *et al*., 2001]. It has been proved that the human subject feels pain even at pH 7.0, which is low enough for the activation of ASIC1a and ASIC3 [Steen, K.H. *et al*., 1995].

We present candidate ASIC genes related to pain and nonpain conditions, gene-gene interactions with the score and importantly their significant pathways. We have proposed a novel Genetic Algorithm approach to optimize gene clusters specific to signaling pathways based on global interaction score. STRING 9.1 tool and related databases have been used for interaction network, score, and primitive cluster.

5.3 Materials and methods

We first described the methods adopted for gene set ranking, gene expression aggregation, and for classifier learning. The input of our experimental workflow was a set of gene expression samples possibly measured by different microarray platforms. To each sample are assigned two labels. The first identifies the microarray platform from which the sample originates; the second identifies a sample class. The output is a classification model, that is, a model that estimates the sample class given an expression sample and its platform label. The model is obviously applicable to any sample not present in the input (“training") data, as

long as its platform label is also known. The remarkable property of the output model used in our approach was instead of combination of separate models each pertaining to a single platform, rather, there was a single classifier trained from the entire heterogeneous sample set and represented in terms of activity scores of units that apply to all platforms, albeit the computation of these activity scores may be different across platforms. More specifically, the activity score of a gene set (such as a pathway) was calculated using a different gene set in each platform. We now describe the individual steps of the method in more detail.

5.3.1 Gene set ranking

Three methods are considered for ranking gene sets. As inputs, all of the methods take a set $G = \{g1, g2,...gp\}$ of interrogated genes, and a set S of N expression samples where for each $si \in S$, $si = (e1,i, e2,i,...ep,i) \in Rp$ where ej,i denotes the (normalized) expression of gene gj in sample si. The sample set S was partitioned into phenotype classes $S = C1 \cup C2 \cup ... \cup Co$ so that $Ci \cap Cj = \{\}$ for $i \neq j$. To simplify this, we assumed binary classification, i.e. $o = 2$. A further input is a collection of gene sets $\mathcal{G}$ such that for each $\Gamma \in G$ it holds $\Gamma \subseteq G$. In the output, each of the methods ranks all gene sets in $\mathcal{G}$ by their estimated power to discriminate samples into the predefined classes.

Next we give a brief account of the three methods and refer to the original sources for a more detailed description.

5.3.2 Gene Set Enrichment Analysis (GSEA)

It tests a null hypothesis that gene rankings in a gene set Γ, according to an association measure with the phenotype, are randomly distributed over the rankings of all genes. It first sorts G by correlation with binary phenotype. Then it

calculates an enrichment score (ES) for each $\Gamma \in G$ by walking down the sorted gene list, increasing a running-sum statistic when encountering a gene $gi \in \Gamma$ and decreasing it otherwise. The magnitude of the change depends on the correlation of *gi* with the phenotype. The enrichment score is the maximum deviation from zero encountered in the random walk. It corresponds to a weighted Kolmogorov-Smirnov-like statistic. The statistical significance of the ES is estimated by an empirical phenotype-based permutation test procedure that preserves the correlation structure of the gene expression data. GSEA was one of the first specialized gene-set analysis techniques. It has been reported to attribute statistical significance to gene sets that have no gene associated with the phenotype, and to have less power than other recent test statistics.

5.3.3 Proposed Framework:

The genes contributing to pain and no pain were taken from the NCBI genes database of ASIC genes. A total of thirty genes were taken out of which eleven were pain genes whereas nineteen were non-pain genes. The next stage is to develop gene-gene interaction network using STRING 9.1 tool. The interaction network is represented in three different views namely confidence, action and evidence view. Using Enrichnet tool, the gene-gene interaction pathways were designed and their p-value was obtained. Significant biological, cellular and molecular components related to candidate genes were found out. The next step involved generating the gene interaction score and storing it in FASTA format as it is the form which is accepted by PROFEAT server as an input. After this pseudo amino acid composition (PAAC) features was generated using PROFEAT server. Once the features of both the pain and non-pain genes were obtained, they were used as the training and testing data to train the classification model. Various

machine learning algorithms were used and their output correlated to predict the pain and nonpain genes.

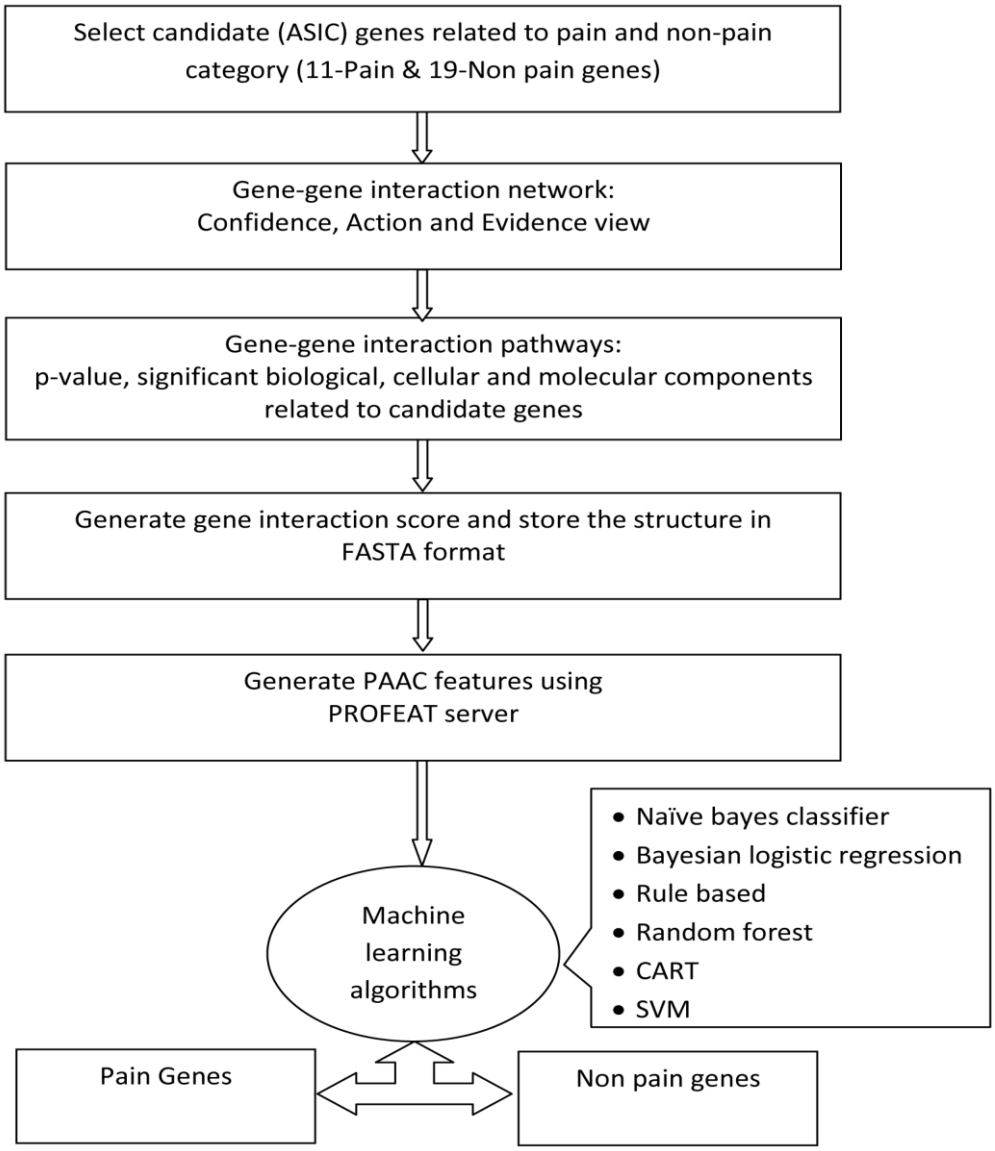

Figure 5.2: Proposed framework for classification of pain and non pain genes using machine learning algorithms.

5.3.4 ASIC genes related to pain and non pain

The candidate gene approach deal with sets of genes in biologically meaningful candidate pathways. Human homologues of genes with well-established molecular and biological functions in synaptic plasticity led to the identification of gene cluster associated with pain sensation. This gene cluster represented

important pain-related molecules such as adenylyl cyclases, kinases, and glutamate receptors. An aggregate, individual gene score based on the gene cluster was also associated with activations in peripheral and central brain regions. Experimental work in animals has shown that pain sensation depends on a cascade of molecular events. Inhibiting ASICs has been shown to relieve pain in a variety of pain syndromes in both animals and humans [Wemmie, J.A. *et al.*, 2006; Dube, G.R. *et al*., 2009; McCleskey, E.W. 1999].

Table 5.1: ASIC pain genes

Gene Name	Description
ASIC1	acid sensing (proton gated) ion channel 1
Asic2	acid sensing (proton gated) ion channel 2
Asic3	acid sensing (proton gated) ion channel 3
ASIC4	acid sensing (proton gated) ion channel family member 4
Kcnk9	potassium channel, subfamily K, member 9
Trpv1	transient receptor potential cation channel, subfamily V, member 1
CFTR	cystic fibrosis transmembrane conductance regulator (ATP-binding)
STOM	Stomatin
DLG4	discs, large homolog 4 (Drosophila)
Trpv4	transient receptor potential cation channel, subfamily V, member 4
Pick1	Protein interacting with C kinase 1

Table 5.2: Non pain genes

Gene Name	Description
TNF	tumor necrosis factor
CFTR	cystic fibrosis transmembrane conductance regulator (ATP-binding)
NGF	nerve growth factor (beta polypeptide)
HLA-DRB1	major histocompatibility complex, class II, DR beta 1
COMT	catechol-O-methyltransferase

ADIPOQ	adiponectin, C1Q and collagen domain containing
MMP3	matrix metallopeptidase 3
HFE	Hemochromatosis
IL10	interleukin 10
PTGS2	prostaglandin-endoperoxide synthase 2 (prostaglandin G/H)
CRP	C-reactive protein, pentraxin-related
IL6	interleukin 6
EDN1	endothelin 1
VEGFA	vascular endothelial growth factor A
SLC6A4	solute carrier family 6 (neurotransmitter transporter), member 4
PEBP1	phosphatidylethanolamine binding protein 1
TGFB1	transforming growth factor, beta 1
BDNF	brain-derived neurotrophic factor
CFTR	Protein coding

Human pain perception is related to variability in genes encoding proteins of the signaling cascade. The individual profile of genetic variability in these signaling molecules correlated significantly with pain. The table 5.1 and 5.2 indicates genes and variability in the human homologues of pain (ASIC) and nonpain signaling genes contributing to inter-individual differences in human pain perception and brain activations.

Thus, the genes described herein appear to form a clusterification with a strong impact on pain performance. Here we found that genes ASIC1,ASIC2, ASIC3, ASIC4, Kcnk9, Trpv1, CFTR, STOM, DLG4, Trpv4 and Pick1 are pain receptor genes whereas TNF, CFTR, NGF, HLA-DRB1, COMT, ADIPOQ, MMP3, HFE, IL10, PTGS2, CRP, IL6, EDN1, VEGF4, SLC6A4, PEBP1, TGFB1 and BDNF are nonpain receptor genes that were considered. All these genes play a significant role in pain and nonpain sensation.

5.3.5 Gene-gene interaction network

Gene interactions are crucial components of all cellular, molecular, biological processes and signaling pathways related to a gene group. Recently, high-throughput methods have been developed to obtain a global description of the interactome (the whole network of gene/protein interactions for a given organism). This estimate was based on the integration of data sets obtained by various methods (mass spectrometry, two-hybrid methods, genetic studies).

The network of interactions between genes is generally represented as an interaction graph, where nodes represent genes and edges represent pair-wise interactions. Graph theory approaches have been applied to describe the topological properties of the network: distribution of node degree (number of incoming and outgoing edges per node), network diameter (average of the shortest distance between pairs of nodes), and clustering coefficient (proportion of the potential edges between the neighbors of a node that are effectively observed in the graph). These analyses have led to the observation of some apparently recurrent properties of biological networks: power-law degree distribution, small world, high clustering coefficients, and modularity.

5.3.6 Gene-gene interaction network in confidence, action and evidence view

The network and scores are accessed from STRING 9.1 tool and database. This tool is used to find interaction network, scores, significant biological, cellular, molecular processes and mapped input genes to these processes and pathways.

In the gene-gene interaction network (graph) the nodes show the input genes mapping and the edges connecting them show interaction. The thicker the edge, higher is the interaction and vise-versa. The interactions are shown in three views

i.e. confidence, action and evidence view having overall, action specific and evidence specific scoring edges.

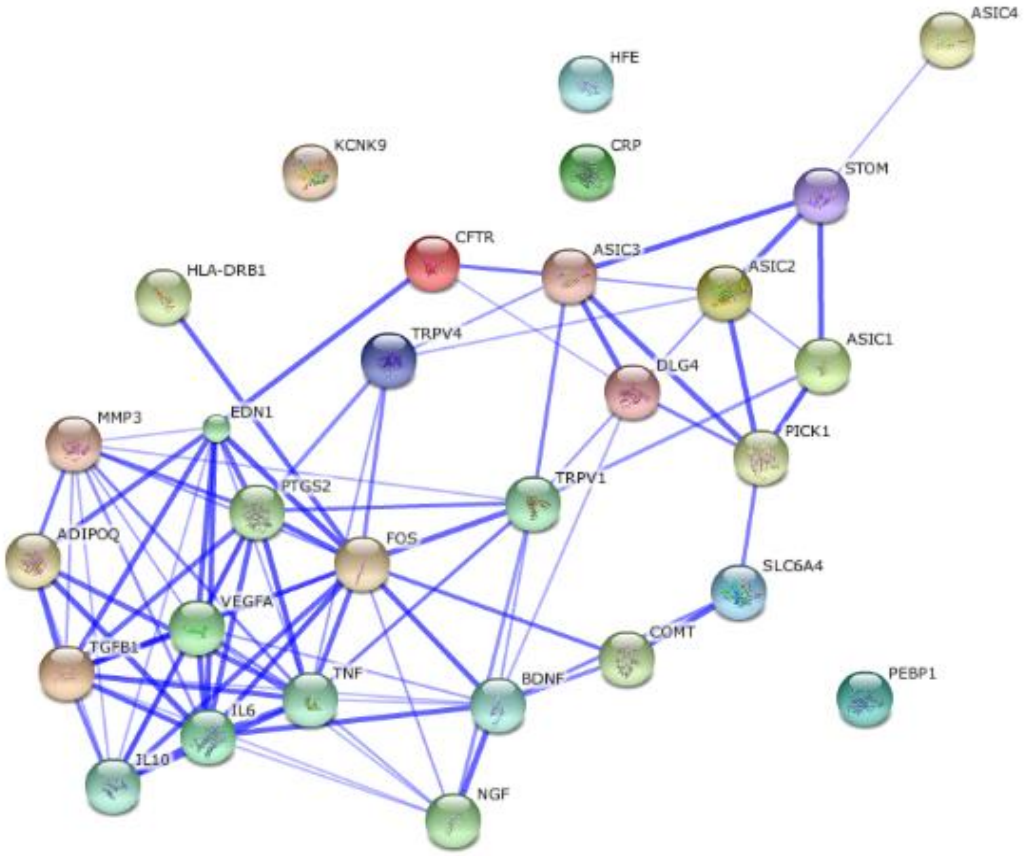

Figure 5.3 (a): Gene-Gene interaction Confidence View

Network Display - Nodes are either colored (if they are directly linked to the input as in the table) or white (nodes of a higher iteration/depth). Edges, i.e. predicted functional links, consist of up to eight lines: one color for each type of evidence.

The confidence view shows overall interaction score corresponding to the neighborhood, gene fusion, co-occurrence, homology, co-expression, experiments, databases and text mining. The individual parameter interaction scores are between [0 1]. For global (overall) interaction score, all individual parameter specific scores are added and normalized between [0 1]. Global interaction scores for gene interaction is not unidirectional, rather it is

bidirectional (Score Adjacency matrix is not symmetric) i.e. score for edge gene(i) to gene(j) is not equal to gene(j) to gene(i). But in the graph, it is represented by adding both.

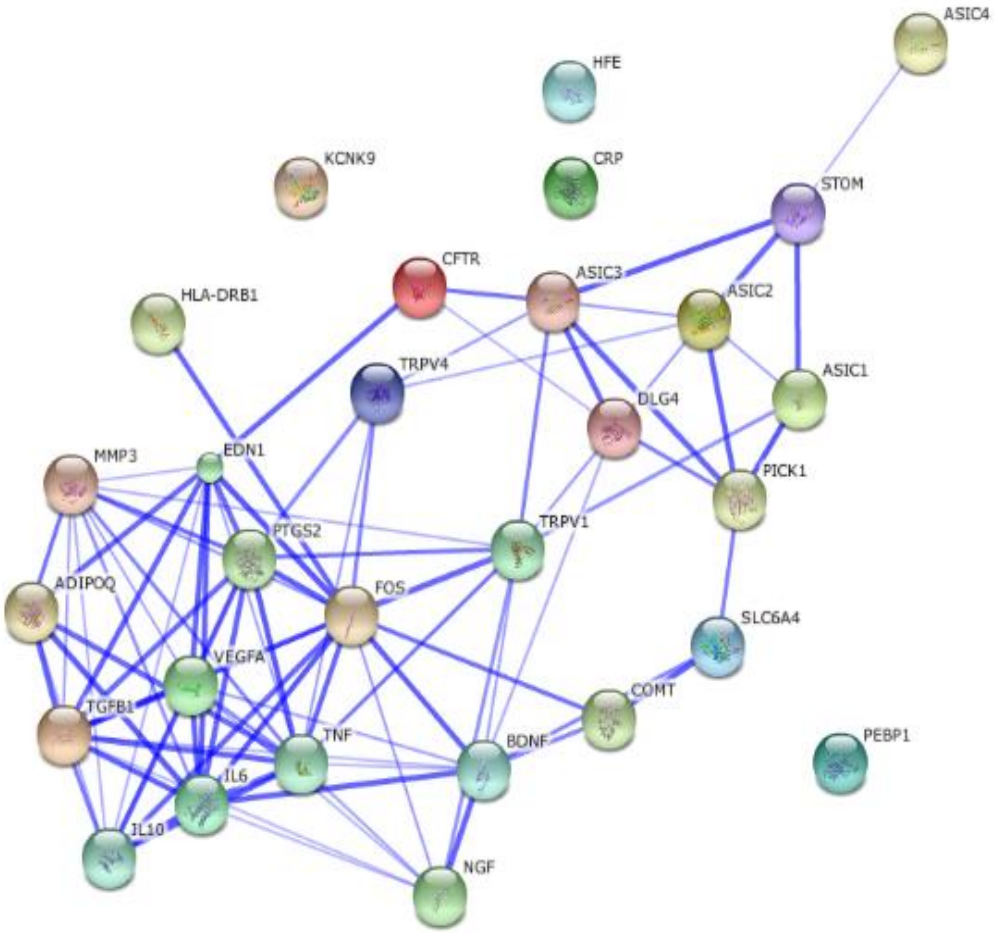

Figure 5.3 (b): Action View – Modes of action are shown in different colors

Likewise, interactions of genes in action and evidence view are shown in Figure 5.3(b) and 5.3(c). Action view interactions are based on activation, inhibition, binding, phenotype, catalysis, post-translation mechanism, reaction, and expression. Evidence view shows the interactions based on neighborhood, gene fusion, co-occurrence, homology, co-expression, experiments, databases and text

mining individually and aggregated and normalized score view results in confidence view.

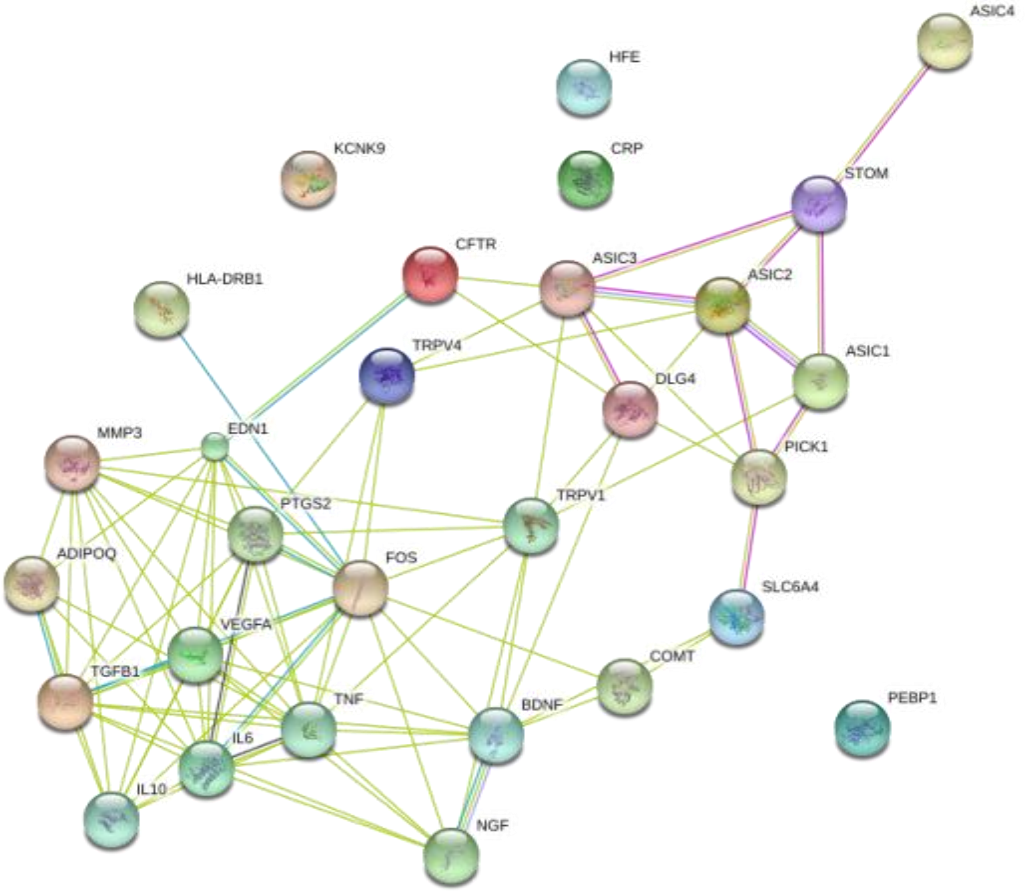

Figure 5.3 (c): Evidence View - Different line colors represent the types of evidence for the association

5.3.7 Gene-gene Interaction Pathways

Assessing functional associations between an experimentally derived gene and protein set of interest and a database of known gene/protein sets is a common task in the analysis of large-scale functional genomics data. For this purpose, a frequently used approach is to apply an over-representation-based enrichment analysis. However, this approach has four drawbacks: (i) it can only score functional associations of overlapping gene/proteins sets; (ii) it disregards genes with missing annotations; (iii) it does not take into account the network structure

of physical interactions between the gene/protein sets of interest and (iv) tissue-specific gene/protein set associations cannot be recognized.

To address these limitations, we used an integrative analysis approach called EnrichNet. It combines a novel graph-based statistic with an interactive sub-network visualization to accomplish two complementary goals: improving the prioritization of putative functional gene/protein set associations by exploiting information from molecular interaction networks and tissue-specific gene expression data and enabling a direct biological interpretation of the results. By using the approach to analyze sets of genes with known involvement in human diseases, new pathway associations are identified, reflecting a dense sub-network of interactions between their corresponding proteins.

5.4 Machine learning

Machine learning deals with artificial systems learning from given data in an autonomous manner. Regarding the available mass of gene expression data with their amount and extent ranging beyond the capacity of any living organism, machine learning proposes an interesting alternative for data analysis and knowledge discovery. Here, we focus on supervised machine learning. From the molecular and biology viewpoint, it provides a form of data analysis going beyond the mere identification of differentially expressed genes or gene sets; particularly, it provides tools designed to solve the problem of inferring a function from data samples and their labels automatically.

In gene-gene interaction and classification, as in many other areas, machine learning is essential since, without machine learning, it would be difficult to work with vast amounts of fuzzy data. In this, a variety of machine learning methods

has been used to classify extracted data. Machine learning is divided into three groups: supervised, unsupervised and semi-supervised learning methods.

Supervised machine learning is a type of machine learning where each sample in the dataset is labeled. The classifier uses a training set to learn a set of parameters and tries to classify the testing set successfully using the learned parameters. Unsupervised machine learning methods try to find hidden structures in an unlabeled data. Semi-supervised machine learning uses both labeled and unlabeled data. A commonly conducted practice in semi-supervised machine learning is to use a small amount of labeled and large amount of unlabeled data for training, and a large amount of labeled and small amount of unlabeled data for testing.

In gene-gene interaction and classification the most commonly used machine learning methods are supervised learning methods such as; SVMs, NNs, and regression analysis. All supervised machine learning methods use a training data to train a hypothesis function for future predictions h (Θ). Training data is defined as such:

$$S = \{(X_i , Y_i) \mid \forall i \in \{ 1, \dots , N\} \} \qquad (5.1)$$

Where, S is the training dataset, X_i is the extracted features, Y_i is the classification of the i^{th} member of the training data and N is the number of subjects for training the hypothesis function h (Θ), where (Θ) = { [Θi] $\forall i \in$ {1,…,N}} for future predictions.

5.4.1 Machine learning for gene expression analysis

Machine learning techniques have been explored [Eisen, *et al*., 1998]. Especially successful were supervised methods (for class prediction or regression) and unsupervised algorithms (for clustering). Gene expression data combined with

machine learning methods revolutionized cancer classification which had been based solely on morphological appearance.

An important milestone was a successful demonstration of cancer classification based solely on high-throughput gene expression data [Golub *et al.,* 1999]. Golub *et al*., (1999) used class discovery and class prediction techniques on acute myeloid leukemia and acute lymphoblastic leukemia microarray data in order to distinguish between the two cancer types using the data without any additional knowledge and to derive a class predictor able to determine the leukemia class for a new unseen case, respectively. While the clustering on samples in the above-mentioned case was used, the clustering on gene level also provides an important insight in analyzed gene expression data; these algorithms have manifested the ability to find groups of co-expressed genes with similar functions which make the clustering algorithms simple but useful tools for gaining leads to gene functions with missing or unavailable functional description [Eisen *et al.,* 1998].

Other methods can still be considered due to their properties, e.g., a natural way to include biological knowledge that improves classifiers interpretability. The forms of the learned classifiers can range from (fast learning and less interpretable) geometrically conceived models such as Support Vector Machines [Cortes and Vapnik, 1995], which have been especially popular in the gene expression domain, to (slower learning and easily interpretable) symbolic models such as logical rules or decision trees that have also been applied in this area.

5.4.2 Naïve Bayes Classifiers

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. These classifiers are highly scalable, requiring a number of

parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. A sample was classified into the class that was most probable given the sample's feature values, according to a conditional probability distribution learned from training data on the simplifying assumption that, within each class, all features are mutually independent random variables. Gene expression data usually deviate from this assumption and consequently the method becomes suboptimal.

5.4.3 Support Vector Machines

A support vector machine is a non-probabilistic binary linear classifier. The main idea behind SVMs is the creation of distinct borders between partitions of given data, inorder to break the data into multiple sections that could be used for classification purposes with the future input [Burges & Christopher, J.C., 1998]. Support vector machines are trained to produce a function that can predict the classification of the future data. Support vector machines are margin optimization models that can classify non-linear data using hyperplanes, rather than greedy output search systems. They use the dataset *S* to train the hypothesis function, h (θ), for future predictions. However, their methodology is a bit different from other methods since SVMs are used to classify data between already known clusters. Support vector machines initially determine the support vectors which are the border elements of a cluster. Then a hyperplane equation was derived using these support vectors.

The following equation was solved for the hyperplane parameters.

$$Xs * W - b = \{ Ys \mid \forall X, W, Y \} \quad (5.2)$$

Where, W is the set of normal vectors that is defined as $W = \{Wi\ (x,y) \mid \forall i \in \{1, \dots, t\}\}$, Xs is the s^{th} support vector, b is the constant in the hyperplane equation, Y is the solution set for the equation.

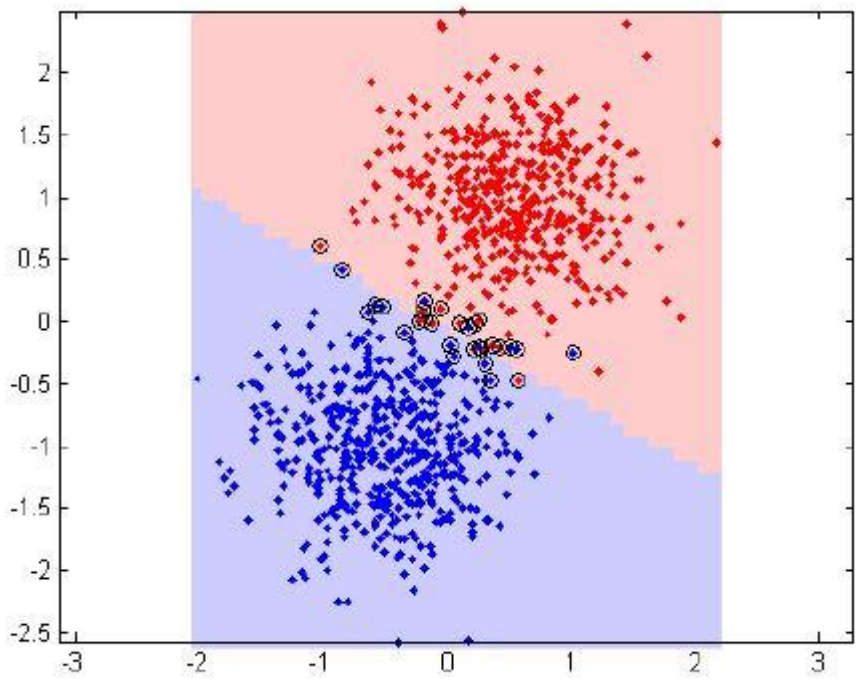

Figure 5.4: Illustrates the use of an SVM in a classification task.

5.4.4 Bayesian Logistic Regression

Classical approaches to classification have been mainly via discriminant rules which are inherently Bayesian in some sense. For example, consider the Bayes rule under a symmetric loss function. Given class conditional densities P_c (**X**) = P (**X**I**Y** = c) of the features **X** in class c and class priors π_c , the posterior probability P (Y = c **I X**) of class c given feature vector **X** is

$$P\ (Y = c\ I\ \mathbf{X}\ ) = \frac{\pi_c\ P_c\ (\mathbf{X})}{\sum_k \pi_k P_k\ (\mathbf{X})} \quad (5.3)$$

Regression analysis is a statistical process for estimating relations between different parts of data. There are different types of Regression Analysis such as the linear approach, the logistic approach, and the ordinary least squares approach. Regression analysis trains a hypothesis function h(Θ) that predicts the probability of Y given X; that is, h(Θ, X) = P(Y | X). Linear regression is the most basic regression analysis approach (Seber et al., 2012). Logistic regression uses the training data S, then trains either a convex or concave estimation function of the form

$$h(\theta, X) = \frac{1}{1+e(\Theta 0+\Theta i+1*Xi)} \quad (5.4)$$

It then compares the result to Yi and uses the success of the estimation to update the parameters of the estimation function [Hosmer, J., 2013].

5.4.5 Rule-based algorithm (decision table)

A decision tree was created in ID3 based on rules which are related to the choice of attributes. There are several algorithms which are based on ID3, such as NewID, CN2, C4.5, and PRISM. The C4.5 provides some additional capabilities which are being ignored in traversing a decision tree whereas PRISM focuses on extracting only the relevant attributes and creates its own combined attribute, unlike ID3. The redundancy is removed by using AQ15 from the initial rule set while NewID supports the attributes that are structured. CN2 incorporates the properties of both AQ15 and ID3, used to select and improve the quality of rules by evaluating it. The space for possible rules was searched by ITURTLE which establishes a ranking based on information content. The CART algorithm seeks to consider the most significant variables discarding the least ones. Lastly, ILLM is used to find the minimal logic expression representing the maximum cases of the

initial rules set. The advantageous part of the decision tree is with the clear understanding of their classification system.

5.4.6 Random Forests (RF)

An RF is a collection of individual decision-tree classifiers, where each tree in the forest has been trained using a bootstrap sample of instances from the data, and each split attribute in the tree is chosen from among a random subset of attributes [Breiman, L., 2001]. Classification of instances is based upon aggregate voting over all trees in the forest. Individual trees are constructed as follows from data having *N* samples and *M* explanatory attributes:

1. Choose a training set by selecting *N* samples, with replacement, from the data.

2. At each node in the tree, randomly select *m* attributes from the entire set of *M* attributes in the data (the magnitude of *m* is constant throughout the forest building).

3. Choose the best split at that node from among the *m* attributes.

4. Iterate the second and third steps until the tree is fully grown (no pruning).

Repetition of this algorithm yields a forest of trees, each of which has been trained on bootstrap samples of instances. Thus, for a given tree, certain instances will have been left out during training. The prediction error is estimated from these 'out-of-bag' instances. The out-of-bag instances are also used to estimate the importance of particular attributes via permutation testing. If randomly permuting values of a particular attribute does not affect the predictive ability of trees on out-of-bag samples, that attribute is assigned a low importance score [Bureau, A. *et al*., 2005]. The decision trees comprising an RF provide an explicit representation of attribute interaction [Breiman, L. 1984] that is readily applicable to the study of

gene-gene or gene-environment interactions. These models may uncover interactions among genes and/or environmental factors that do not exhibit strong marginal effects. Additionally, tree methods are suited to dealing with certain types of genetic heterogeneity, since early splits in the tree define separate model subsets in the data. RFs capitalize on the benefits of decision trees and have demonstrated excellent predictive performance when the forest is diverse (i.e. trees are not highly correlated with each other) and composed of individually strong classifier trees. The RF method is a natural approach for studying gene-gene or gene-environment interactions because importance scores for particular attributes take interactions into account without demanding a pre-specified model [Lunetta, K.L. *et al*., 2004].

However, most current implementations of the importance score are calculated in the context of all other attributes in the model. Therefore, assessing the interactions between particular sets of attributes must be done through careful model interpretation, although there has been a preliminary success in jointly permuting explicit sets of attributes to capture their interactive effects. In selecting functional SNP attributes from simulated case-control data, RFs outperform traditional methods such as the Fisher's Exact test when the 'risk' SNPs interact and the relative superiority of the RF method increases as more interacting SNPs are added to the model. RFs have also shown to be more robust in the presence of noise SNPs relative to the methods that rely on main effects, such as the Fisher's Exact test. It is anticipated that RFs will prove a useful tool for detecting gene-gene interactions.

5.4.7 Neural Networks (NN)

NNs are a popular machine-learning model based on the brain's ability to solve problems. The terminology of machine learning itself is inspired by this model

that is why the term ‘learn’, as opposed to ‘fit’, is used to describe the process of finding a model that describes some data. An NN can be thought of as a directed

graph composed of nodes that represent the processing elements (or neurons), arcs that represent the connections of the nodes (or synaptic connections), and directionality on the arcs that represent the flow of information, as illustrated in figure 5.5.

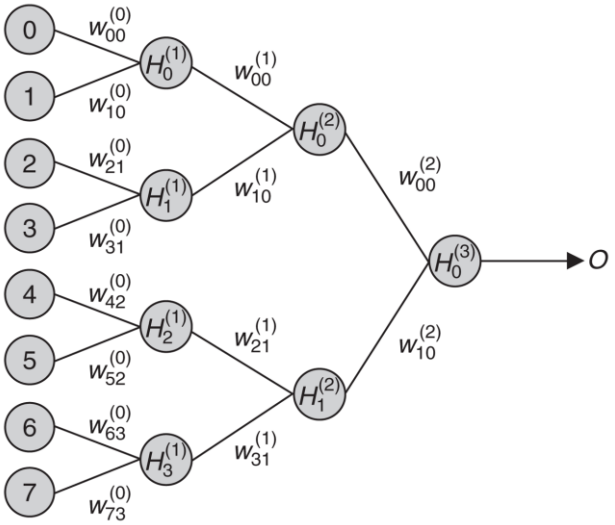

Figure 5.5: Example of a backpropagation neural network with eight input nodes (*Xi*) and three hidden layers with four nodes in the first layer, two nodes in the second layer and one node in the third layer. The signal is propagated through the network to yield an output signal (*O*), which can be put to a threshold to yield an output for affected (case) or unaffected (control). $\boldsymbol{Hj^{(k)}}$ = value of node *j* in layer *k*; $\boldsymbol{wji^{(k)}}$ = weight of the connection between node *j* in layer *k* with node *i* in layer *k*+1.

The processing elements, or nodes, are arranged in layers. The input layer receives an external pattern vector for processing. Each node (*Xi*) in the input layer is then connected to one or more nodes in the first hidden layer ($Hj^{(1)}$). The

nodes in the first hidden layer are in turn connected to nodes in additional hidden layers or to each output node (*O*). The number of hidden layers can range from zero to as many as is computationally feasible. Each network connection has an associated weight, or coefficient, ($w_{ji}^{(k)}$). The signal is conducted from the input layer through the hidden layers to the output layer. The output layer, which often consists of a single node, generates an output signal that is used to classify the input pattern [Skapura, D. 1995]. While NNs are often considered to be a mysterious black box, they are really a series of nonlinear statistical models, similar to regression models. NNs can be expressed as a weighted linear combination of inputs. Each hidden node can be represented as a weighted sum of its inputs. For example, the output from the input nodes to the nodes in the first hidden layer can be written as (equation 5.1):

$$H_j^{(1)} = \sigma\left(\sum_i w_{ji}^{(0)} X_i\right) \qquad \ldots\ldots \text{ (Eq. 5.5)}$$

where σ is a nonlinear activation function, usually chosen to be sigmoid $1/(1 + e^{-x})$, and $w_{ji}^{(0)}$ are the weights for the connections between input nodes *Xi* and nodes $H_j^{(1)}$ in hidden layer 1. The output for nodes in subsequently hidden layers (*k*) can be written as a recurrence relation between the previous hidden layer nodes (equation 5.6):

$$H_j^{(k)} = \sigma\left(\sum_i w_{ji}^{(k-1)} H_i^{(k-1)}\right) \qquad \ldots\ldots \text{(Eq. 5.6)}$$

and the target output can then be modelled as a linear combination of the hidden layers (equation 5.7):

$$O = \sum_{jk} H_j^{(k)} \qquad \ldots\ldots \text{(Eq. 5.7)}$$

The input pattern vector that is propagated through the network can consist of continuous or discrete values. This is also true of the output signal. Designing the network architecture must take into account the representation of the input pattern vector and it's interaction with the network while propagating information through the network. Thus, the data representation scheme must be suitable to detect the features of the input pattern vector so that it produces the correct output signal. A large field of neural network design has been devoted to the question of proper data representation.

Since learning and memory are thought to be associated with the strength of the synapse, setting the strength of the NN connections (or synaptic weights) is the mechanism that allows the network to learn [Tarassenko, L. 1998]. The connection strengths, together with their inputs, lead to an activity level, which is then used as input for the next layer of the NN (Anderson, J. 1995). NNs often function with backpropagation types of error minimisation functions, also called gradient descent. Since learning is associated with the synaptic weights, backpropagation algorithms minimise the error by changing the weights following each pass through the network. This 'hill-climbing' algorithm makes small changes to the weights until it reaches a value to which any change makes the error higher, indicating that the error has been minimised. Several research groups have used NNs for genetic studies because of their potential for detecting gene-gene or gene-environment interactions in addition to main effects [Bhat., A. 1999; Bicciato, S. *et al*., 2003].

These studies had varying levels of success because of the challenges associated with designing the appropriate NN architecture. Ritchie *et al.,* (2003) proposed a novel NN technique that uses evolutionary algorithms to optimise both the inputs

and the architecture of NNs. GP was used to optimise the NN architectures, where each GP binary expression tree represents an NN. The GP-optimised NN (GPNN) optimises the inputs from a larger pool of variables, the weights, and the connectivity of the network, including the number of hidden layers and the number of nodes in the hidden layer. Thus, the algorithm attempts to generate the appropriate network architecture for a given dataset.

To evaluate the ability to detect gene-gene interactions, Ritchie *et al*., (2003) performed simulation studies in which data were simulated from a set of different models exhibiting gene-gene interactions (epistasis). Five different epistasis models were simulated, each exhibiting interactions between two genes. To determine if the addition of the evolutionary algorithm increased the power of the NN for detecting interactions, GPNN was compared with a traditional back propagation NN (BPNN). Two different analyses were conducted to compare the performance of the BPNN and the GPNN. Firstly, the ability to model gene-gene interactions was determined by comparing the classification and prediction error of the two methods using data containing only the two interacting genes. Secondly, the ability to detect and model gene-gene interactions was investigated for both NN approaches. This was determined by comparing the classification and prediction errors of the two methods using data containing the interacting genes and a set of other non-functional genes. GPNN was able to model nonlinear interactions as well as a traditional BPNN based on the analyses of only the interacting genes. In addition, GPNN had improved power and predictive ability compared with BPNN when applied to data containing both functional and non-functional genes. These results provided evidence that GPNN is able to detect the functional SNPs and model the interactions for the epistasis models described.

5.5 Result

Table 5.3: Displays the various machine learning algorithms used to classify the pain and non-pain genes along with their accuracy.

Machine Learning Algorithms	Correctly classified instances	Incorrectly classified instances	Accuracy (%)
Naive Bayes classifier	23	07	76.66
Support vector machine	25	05	83.33
Bayesian logistic regression	19	11	63.33
Rule Based	21	09	70
Random Forest	15	15	50
Neural Network	22	08	73.33
CART	19	11	63.33

*10 fold cross validation is used to extract the results

We experimented with seven diverse machine learning algorithms to avoid dependence of experimental results on a specific choice of a learning method. In experiments, we used the implementations available in the WEKA software. SVM prevails in predictive modeling of gene expression data giving an accuracy rate of 83.33% followed by Naïve bayes classifier with 76.66%. However SVM is usually associated with high resistance to noise in data.